УДК 004.83

ДОСЛІДЖЕННЯ БАЗИ ЗНАНЬ З АРХІТЕКТУРИ ІНФОРМАЦІЙНИХ СИСТЕМ

ДЕНІСОВА О. О.
кандидат економічних наук
КИЇВ

Постановка проблеми. Архітектурний підхід до розробки інформаційних систем є одним з перспективних способів забезпечення ефективного інноваційного використання інформаційних технологій для трансформації бізнесу з метою виконання вимог сучасної розвиненої ринкової економіки. Його відмітною особливістю є оперування невимірюваними змінними, нечисловими інструментами, практичним досвідом. Робота з архітектурою спрямовується на виконання вимог клієнта на відміну від інжинірингу, зорієнтованого на технічну оптимізацію. У цьому контексті першочерговим завданням є створення і використання бази знань з архітектури інформаційних систем.

Аналіз останніх досліджень і публікацій. До організації знань з архітектури нині існує чотири підходи [1], кожний з яких виокремлює свої специфічні будівельні блоки і на практиці доповнює інші:

– підхід, зорієнтований на паттерни – задокументовані перевірені рішення повторюваних проблем. Якщо у середині 90-х років XX ст. паттерни були популярним рішенням полегшення об'єктно-орієнтованого проектування (прикладом є паттерни групи Gang of Four), нині існують паттерни для аналізу, архітектурного проектування [2, 3] і процесу розробки ІС. Під час проектування архітектури паттерни використовуються як словники, що полегшують обмін знаннями між розробниками. Хоча найчастіше паттерни документуються згідно певних шаблонів, єдиний стандарт в цій галузі відсутній. Це ускладнює розуміння і використання готових паттернів, а їх автоматизоване оброблення залишається перспективою;

– підхід, зорієнтований на динамізм архітектури, передбачає автоматизований пошук відображення системи і виведення висновків щодо її модернізації. Використовуються формалізовані знання, як правило, у формі графів, що подають компоненти і конектори, а реконфігурація виражається через правила переформування графів. Інші підходи, в яких графові структури є неявними, застосовують алгебру процесів або логічні правила. Окремим сімейством формальних мов для подання є архітектур є спеціалізовані мови з загальною назвою ADL (architecture description languages, мови опису архітектур) – ACME, Wright, C2, Darwin, Koala, AADL. Ці мови відрізняє зорієнтованість на подання архітектури програмного забезпечення. Для моделювання підприємств розроблено інші спеціалізовані мови, які не обов'язково передбачають розгляд компонентів програмного забезпечення, –ArchiMate, DEMO, ABACUS. Різноманітність поглядів на застосування мов ADL, складність їх розбору і відсутність автоматизованої підтримки зумовлюють нехтування цим підходом у реальних проектах;

– підхід, зорієнтований на вимоги. Сучасний підхід до проектування передбачає спільну розробку вимог і проектних рішень – вимоги визначають рішення, але без знання можливих рішень зацікавлені особи часто не можуть сформулювати свої вимоги до майбутньої системи. Прикладами відповідних методологій є архітектурні фрейми Джексона, цілеспрямований інжиніринг вимог, модель з двома максимумами для з'єднання архітектури і вимог [4];

– підхід, зорієнтований на рішення. У той час як існуючі структури (TOGAF, E2AF, FEAF та ін.) подають результати проектування на досить високому рівні абстракції, у більшості випадків залишається невизначеною раціональна база проектних рішень – причини їх прийняття і логічні міркування з їх обґрунтування.

Формулювання цілей статті (постановка завдання). Прийняття архітектурних рішень передбачає розгляд різноманітних аспектів і проблем, зокрема технічних, бізнесових, політичних і соціальних з залученням багатьох учасників і просто зацікавлених осіб, які обмінюються аргументами, поглядами, ідеями, оцінками тощо. При цьому виникає необхідність нагромадження і роботи з явними та неявними, загальними і специфічними знаннями. Сучасні засоби автоматизованого керування архітектурними знаннями (SEI-ADWiki, ADkwik, ADDSS2, Archium, AREL4, SEURAT5, EAGLE, PAKME та ін.) спрямовані на роботу з явними, формальними знаннями, обмежуючись при цьому документуванням рішень і забезпеченням групового використання і керування архітектурними знаннями. Очевидно, що такий підхід є надто спрощеним і не забезпечує повноцінної підтримки роботи архітектора і проектувальника. Водночас структуру бази архітектурних знань, зокрема її частину, пов'язану з рішеннями, у теоретичних дослідженнях визначено лише частково, що і зумовило завдання даного дослідження.

Виклад основного матеріалу. Структура бази архітектурних знань визначається функціями, що вона їх повинна виконувати у системі керування архітектурою. Головними з них є підтримка визначення архітектурних артефактів і обґрунтування проектних рішень.

Для виконання названих функцій база знань повинна містити такі елементи, що створюються і використовуються протягом аналізу, синтезу, оцінювання, реалізації та підтримки архітектури: проблеми, що потребують вирішення, поставлені цілі, вимоги до інформаційної системи, умови (прийняті припущення, обмеження, аргументи, принципи тощо), паттерни всіх видів, альтернативи і рішення, що приймаються, наслідки, що очікуються або настали, об-

ґрунтування (пояснення, чому було прийнято рішення, в явній формі), компоненти архітектури. Кожен елемент слід описати сукупністю атрибутів, що його повністю характеризують і дають можливість його інтерпретувати. Серед таких атрибутів особливе значення мають параметри використання паттернів, які дають змогу перевірити їх відповідність висунутим вимогам і поставленим цілям, а також встановити можливість їх застосування у конкретній ситуації, та оцінки ваги, терміновості, вартості, ризикованості або ін., що виставляються елементам проектувальниками або іншими зацікавленими особами під час створення елемента або після завершення проекту з метою нагромадження досвіду.

Між елементами встановлюються зв'язки таких типів:

– генеративний – встановлюється між елементами одного типу для виокремлення певних сукупностей, наприклад, функціональних вимог з усіх вимог, що їх було визначено. Особливо корисні генеративні зв'язки для формування обґрунтувань, при цьому вони можуть посилатись на інші зв'язки, формуючи ланцюжки з взаємопов'язаних елементів для безперервності розмірковувань;

– посесивний – зв'язок між цілим і частиною для елементів одного типу;

– дестинативний – визначається призначення елемента, зокрема, рішення приймається для досягнення певної цілі, аналогічно – паттерн, компонент архітектури;

– каузальний та зворотний йому результативний – позначають прийняття рішення внаслідок настання обставин (описуються через обґрунтування), появи вимог, проблем або прийняття інших рішень;

– потенсивний і негативний – доповнюють попередні, подаючи можливість збільшення або виключення ймовірності настання передумов або появи наслідків при настанні інших подій;

– корелятивний, комутативний – визначаються взаємопов'язані елементи, наприклад, рішення, що приймаються сумісно, або компоненти архітектури, що відповідають одне одному;

– репродуктивний, ситуативний, лімітативний – стосуються умов прийняття рішень, параметрів застосування паттернів;

– інструментальний, медитативний – позначення інструментів і способів виконання прийнятого рішення;

– трансгресивний – відображають результати перетворення елементів.

Ситуативність встановлених зв'язків забезпечується вказівкою статусу елемента. Так, рішення можуть визначатись як запропоновані, можливі (такі, що розглядаються), прийняті. Умови і наслідки розглядаються варіативно зі зміною ймовірностей їх настання для дослідження різних сценаріїв розвитку подій. З набору цілей можуть вибиратись ті, що відповідають певному критерію (важливості, терміновості, інтересам зацікавленої особи).

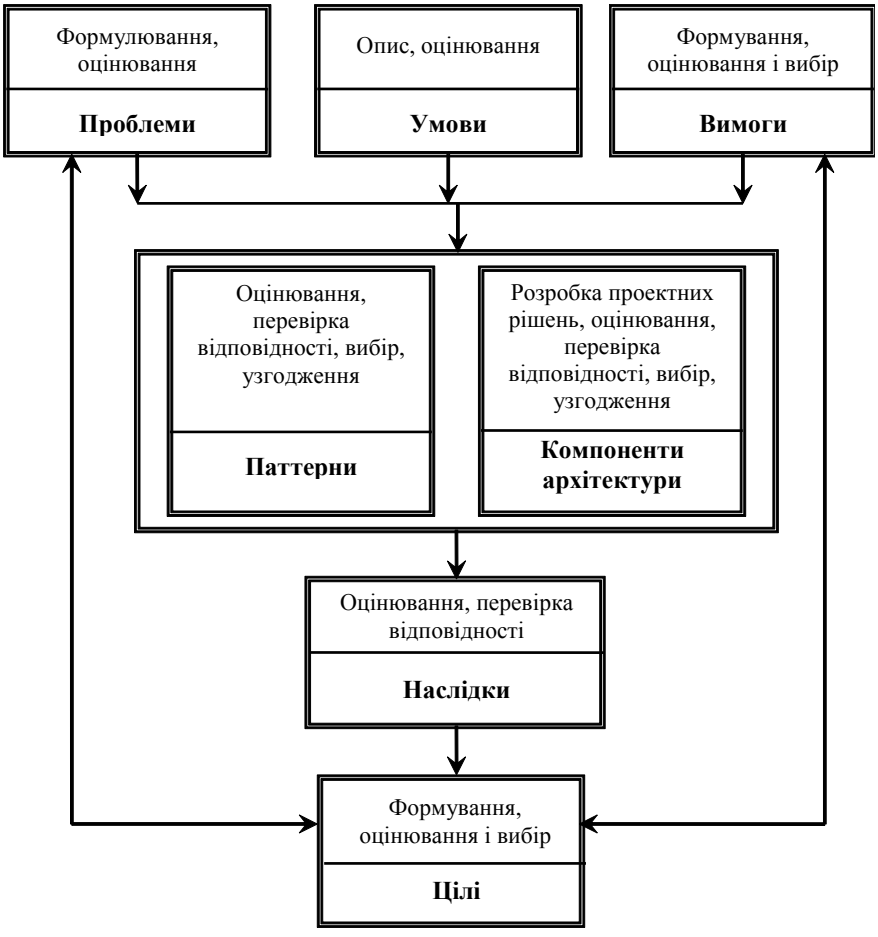

Рис. 1. Схема процесу підтримки прийняття рішень на основі елементів бази архітектурних знань

Наявність різноманітних зв'язків, що утворюють ланцюжки, забезпечує повний контекст прийняття рішень (див. рис. 1). При цьому може бути організована підтримка як класичної моделі прийняття рішень (визначаються альтернативи, критерії їх оцінювання, вага кожного критерію), так і «натуралістичної» (моделі розпізнавання рішень) – параметри рішень/патернів порівнюються з параметрами проблем, вимог та умов прийняття рішень з метою виявлення підхожого. Запропонована структура бази знань надає можливість розширити цю модель – рішення порівнюється за наслідками від його прийняття з поставленими цілями, які, в свою чергу, впливають на формулювання та оцінювання проблем і вимог.

Висновки. Запропонована структура бази знань з архітектури інформаційних систем сполучає різні підходи до організації знань і надає можливість документувати обґрунтування прийнятих проектних рішень, нагромаджувати практичний досвід з керування архітектурою та проводити розмірковування на основі аналогій з метою підвищення якості рішень. ■

ЛІТЕРАТУРА

1. Babar M. A. and oth. Software Architecture Knowledge Management. Theory and Practice. – Springer, 2009. – 279 p.

2. Fowler M. Patterns of Enterprise Application Architecture. – Addison-Wesley, Boston MA, 2003. – 560 p.

3. Buschmann F., Meunier R., Rohnert H., Sommerlad P. Pattern-Oriented Software Architecture: A System of Patterns. – Addison-Wiley, NewYork, 1996. – 456 p.

4. Galster M., Eberlein A., Moussavi M. Transition from requirements to architecture: A review and future perspective// 7th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD), 2006. – pp. 9–16.